DIFFUSION FILTERING OF GRAPH SIGNALS AND ITS USE IN RECOMMENDATION SYSTEMS

Jeremy Ma[†] *Weiyu Huang*[‡] *Santiago Segarra*[‡] *Alejandro Ribeiro*[‡]

[†]School of Electrical Engineering and Telecommunications, University of New South Wales
[‡]Department of Electrical and Systems Engineering, University of Pennsylvania

ABSTRACT

This paper presents diffusion filtering as a method to smooth signals defined on the nodes of a graph or network. Diffusion filtering considers the given signals as initial temperature distributions in the nodes and diffuses heat through the edges of the graph. The filtered signal is determined by the accumulated temperatures over time at each node. We show multiple other interpretations of diffusion filtering and describe how it can be generalized to encompass a wide class of networks making it suitable for real-world applications. We prove that diffused signals are stable to perturbations in the underlying network. Further, we demonstrate how diffusion filtering can be applied to improve the performance of recommendation systems by considering the problem of predicting ratings from a signal processing perspective.

Index Terms— Graph signal processing, filtering, pattern recognitions, recommendation systems.

1. INTRODUCTION

Networks and graphs can be defined as structures that encode pairwise relations between elements of a set. The simplicity of this definition drives the application of graphs and networks to a wide variety of disciplines such as biology [1,2] and sociology [3,4]. The emerging theory of graph signal processing [5] is not specifically concerned with the study of networks but rather with the study of *signals* supported on the nodes of a graph, where the latter captures an underlying notion of proximity between the nodes. Examples of network-supported signals include gene expression patterns defined on top of gene networks [6] and brain activity signals supported on top of brain connectivity networks [7].

In this paper we address the problem of smoothing out irregularities of graph signals by incorporating the structure of the underlying network. Since the network encodes similarities between nodes, we assume that the signals supported on the graph also exhibit such similarities and want the filtered signal to be such that signals on nodes with high proximities become more alike. We present the notion of diffusion filtering and argue that it inherits this functionality through its connection to diffusion processes which, in turn, draw their inspiration from the diffusion of heat in continuous matter [8, 9]. The linear differential equation that models heat diffusion can be extended to encompass dynamics in discrete structures such as networks where every node is interpreted as containing an amount of heat that flows from hot to cold nodes [10, 11]. Heat flows through the edges of the graph and the rate at which heat diffuses is proportional to both the heat imbalance between the nodes adjacent to the edge and the edge weight representing the proximity between these nodes. Diffusion processes are often used to exploit their asymptotic configurations in steady state such as in problems of formation control [12] as well as the propagation of opinions in social networks [13]. In this paper we do not exploit the asymptotic, but rather the transient behavior of diffusion processes.

The main contributions of this paper are the definition and interpretation of diffusion filtering for signals supported on networks (Section 2), its generalization to arbitrary graphs (Section 3), the proof that the diffused signals are stable to perturbations in the underlying network (Section 4), and the utility of diffusion filtering to improve the performance of recommendation systems (Section 5).

2. DIFFUSION FILTERING

Consider weighted and symmetric networks formally defined as triplets $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$. The set $\mathcal{V} = \{1, \ldots, n\}$ represents the n vertices, the set $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ denotes the edges defined as ordered pairs (i, j), and $\mathcal{W} : \mathcal{E} \to \mathbb{R}_{++}$ is a map from the set of edges to the strictly positive reals associating a weight $w_{ij} > 0$ to each edge (i, j). Since the graph is symmetric we have $(i, j) \in \mathcal{E}$ if and only if $(j, i) \in \mathcal{E}$ and $w_{ij} = w_{ji}$ for all $(i, j) \in \mathcal{E}$. The edge $(i, j) \in \mathcal{E}$ symbolizes the existence of a relationship between i and j and we say that i and j are adjacent or neighboring. The set of neighboring nodes of i is denoted as $\mathcal{N}_i$. The weight $w_{ij} = w_{ji}$ stands for the strength of the relation, or, equivalently, the proximity or similarity between nodes i and j. The networks considered here do not contain self loops, i.e., $(i, i) \notin \mathcal{E}$ for all $i \in \mathcal{V}$.

We utilize the customary definitions of adjacency, degree, and Laplacian matrices for the weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$ [14, Chapter 1]. The adjacency matrix $\mathbf{A} \in \mathbb{R}_{+}^{n\times n}$ is defined such that $A_{ij} = w_{ij}$ whenever i and j are adjacent, i.e., whenever $(i, j) \in \mathcal{E}$, with $A_{ij} = 0$ for $(i, j) \notin \mathcal{E}$. The degree matrix $\mathbf{D} \in \mathbb{R}_{+}^{n\times n}$ is a diagonal matrix with its i-th diagonal element $D_{ii} = \sum_j w_{ij}$. The Laplacian matrix is defined as the difference $\mathbf{L} := \mathbf{D} - \mathbf{A} \in \mathbb{R}^{n\times n}$, thus, its elements are explicitly given by

$$L_{ij} = \begin{cases} -A_{ij}, & \text{if } i \neq j, \\ \sum_{k=1}^{n} A_{ik}, & \text{if } i = j. \end{cases} \tag{1}$$

Models with asymmetric weights $w_{ij} \neq w_{ji}$ and signed edges $w_{ij} < 0$ are considered in Section 3.

Our interest in this paper is on signals $\mathbf{x} = [x_1, \ldots, x_n]^T \in \mathbb{R}^n$ whose component x_i is interpreted as a value associated with the ith node of $\mathcal{G}$. For a given graph with Laplacian $\mathbf{L}$ and given signal $\mathbf{x}$, introduce a constant $\alpha > 0$ to define the time-varying vector $\mathbf{x}(t) \in \mathbb{R}^n$ as the solution of the linear differential equation

$$\frac{d\mathbf{x}(t)}{dt} = -\alpha\, \mathbf{L}\, \mathbf{x}(t), \qquad \mathbf{x}(0) = \mathbf{x}. \tag{2}$$

The differential equation in (2) models the diffusion of heat on the graph $\mathcal{G}$ when the weights represent thermal conductivity across the edges [10]. The given vector $\mathbf{x} = \mathbf{x}(0)$ specifies the initial temperature distribution and $\mathbf{x}(t)$ represents the temperature distribution at time t. The constant α scales the thermal conductivities and controls the heat diffusion rate. The solution of (2) is given by the matrix exponential

$$\mathbf{x}(t) = e^{-\alpha\, \mathbf{L}\, t}\, \mathbf{x}. \tag{3}$$

Using (3) we can compute the temperature distribution at any time point t by diffusing the initial heat profile $\mathbf{x}$ on the structure of the underlying network through its Laplacian $\mathbf{L}$. Notice that as diffusion continues, $\mathbf{x}(t)$

Supported by NSF CCF-1217963.

settles to an isothermal equilibrium – temperature becomes identical on all nodes – as long as the graph is fully connected.

It is informative to write a componentwise version of (2). If we focus on the i-th entry $x_i(t)$ of $\mathbf{x}(t)$, it follows from (1) and (2) that

$$\frac{dx_i(t)}{dt} = \sum_{j \in \mathcal{N}_i} \alpha\, w_{ij} \left(x_j(t) - x_i(t) \right). \tag{4}$$

Since (4) describes the rate of change in $x_i(t)$, it follows that the flow of heat on an edge increases proportionally with the temperature imbalance $x_j(t) - x_i(t)$ as well as the weight w_{ij}. In particular, with other things being equal, nodes with larger proximity tend to equalize their temperatures faster. This means that the time varying vector $\mathbf{x}(t)$ defines a series of temperature profiles by smoothing the initial distribution $\mathbf{x}$ using the underlying graph structure. This allows the interpretation of $\mathbf{x}(t)$ as a smoothed version of the given signal $\mathbf{x}$ where variations between nodes that are strongly related are reduced.

While it is possible to focus on a particular $\mathbf{x}(t)$ as a filtered version of $\mathbf{x}$ it is more convenient to aggregate the whole diffusion process. A possible way of constructing a vector $\mathbf{y}$ condensing the time varying vector $\mathbf{x}(t)$ is to consider the discounted integral

$$\mathbf{y} = \int_0^\infty e^{-t}\, \mathbf{x}(t)\, dt = \int_0^\infty e^{-t}\, e^{-\alpha\, \mathbf{L}\, t}\, \mathbf{x}\, dt. \tag{5}$$

We say that the vector $\mathbf{y}$ in (5) is the diffused or diffusion filtered signal obtained from $\mathbf{x}$ associated with $\mathbf{L}$. The motivation for (5) is that it considers the heat diffusion profile generated by $\mathbf{x}$ on $\mathbf{L}$ while giving more importance to initial times. This is necessary because in the asymptotic thermal equilibrium the variability of the original signal is completely smoothed out. Further note that the primitive of the matrix exponential $e^{-t}e^{-\alpha\mathbf{L}t} = e^{-(\mathbf{I}+\alpha\mathbf{L})t}$ is the matrix $-(\mathbf{I}+\alpha\mathbf{L})^{-1}e^{-(\mathbf{I}+\alpha\mathbf{L})t}$ with $\mathbf{I}$ being the identity matrix of corresponding size. This can be used to solve (5) in closed-form as we summarize in the following definition.

Definition 1 *Given a network $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$ with Laplacian $\mathbf{L}$, the diffusion rate $\alpha \geq 0$, and a graph signal $\mathbf{x} \in \mathbb{R}^n$ defined in the node space $\mathcal{V}$, the diffusion filtered signal $\mathbf{y}$ obtained from $\mathbf{x}$ is defined as*

$$\mathbf{y} = \left(\mathbf{I} + \alpha\, \mathbf{L}\right)^{-1} \mathbf{x}. \tag{6}$$

Observe that the diffused signal $\mathbf{y}$ is another graph signal defined in the same node space $\mathcal{V}$ and that $\mathbf{y}$ always exists because $\mathbf{I} + \alpha\mathbf{L}$ is strictly diagonally dominant [15, Thm 6.2.27]. Besides its construction from diffusion dynamics, diffusion filtering can also be seen to follow as the solution of the optimization problem [16, 17],

$$\min_{\mathbf{y} \in \mathbb{R}^n} \|\mathbf{x} - \mathbf{y}\|_2^2 \;+\; \alpha\, \mathbf{y}^T \mathbf{L} \mathbf{y}. \tag{7}$$

The diffused signal is then close to $\mathbf{x}$ but with small total variation on the graph, as indicated by the first and second summands in (7), respectively. The balance of these conflicting requirements is mediated by α. Alternatively, we can also think of $\mathbf{y}$ as the result of processing $\mathbf{x}$ with a low-pass graph filter [18–20]. To explain this interpretation, consider the eigendecomposition of the Laplacian $\mathbf{L} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^H$ and define the graph Fourier transforms of $\mathbf{x}$ and $\mathbf{y}$ as $\hat{\mathbf{x}} = \mathbf{V}^H\mathbf{x}$ and $\hat{\mathbf{y}} = \mathbf{V}^H\mathbf{y}$, respectively. The form of (6) is such that

$$\hat{y}_i = \frac{1}{1 + \alpha\lambda_i} \hat{x}_i. \tag{8}$$

Therefore, higher frequency components of $\mathbf{x}$, which are associated with larger eigenvalues λ_i, are attenuated by a larger coefficient. Conversely, lower frequencies are attenuated by a smaller coefficient as is expected of a (graph) low-pass filter. Moreover, the larger the diffusion rate α the steeper the decay of the graph filter response for high frequencies. These three interpretation – diffusion dynamics, a signal close to the original with smaller total variation, and the result of applying a low-pass filter – are compatible. They all imply that $\mathbf{y}$ is obtained from $\mathbf{x}$ by reducing the level of variability as measured by the graph Laplacian $\mathbf{L}$. The diffusion filtering is closely related with Laplacian smoothing, e.g. [21, 22]. In this paper, we exemplify the use of diffusion filtering in the problem of predicting ratings in a recommendation system.

3. GENERALIZED DIFFUSION FILTERING

The diffusion filter in Section 2 is derived for symmetric networks with positive weights and is defined in terms of the graph Laplacian. In some applications, recommendation systems being an example, we have to deal with asymmetric networks with weights that can be negative and the structure of the problem justifies the use of normalized Laplacians. We discuss these extensions in this section.

Normalized Laplacian. The random walk normalized Laplacian is defined as $\tilde{\mathbf{L}} := \mathbf{D}^{-1}\mathbf{L}$; see, e.g., [23]. In terms of an explicit representation akin to the Laplacian in (1), this definition means that values in the diagonals are $\tilde{L}_{ii} = 1$ and that the off-diagonal values are $\tilde{L}_{ij} = -w_{ij}/\sum_{j\in\mathcal{N}_i} w_{ij}$. To define the normalized diffusion filter we just replace $\mathbf{L}$ with $\tilde{\mathbf{L}}$ in Definition 1. This results in

$$\mathbf{y} = \left(\mathbf{I} + \alpha\, \tilde{\mathbf{L}}\right)^{-1} \mathbf{x} = \left(\mathbf{I} + \alpha\, \mathbf{D}^{-1}\mathbf{L}\right)^{-1} \mathbf{x}. \tag{9}$$

This expression corresponds to diffusion dynamics where the conductances out of a node are normalized to sum up to one. In particular, the componentwise rate of change in node i induced by its neighbors becomes [cf. (4)]

$$\frac{dx_i(t)}{dt} = \Big(\sum_{j \in \mathcal{N}_i} w_{ij} \Big)^{-1} \sum_{j \in \mathcal{N}_i} \alpha\, w_{ij} \left(x_j(t) - x_i(t) \right). \tag{10}$$

In (4), nodes with higher degree change at a faster rate. In (10), the rate of change of x_i is driven by the difference with its neighbors and their *relative* proximity but is independent of the node's degree. This property is of interest when the underlying graph has a heterogenous degree distribution. In practice, we find that normalized diffusion filtering is indeed beneficial in these situations; see Section 5.

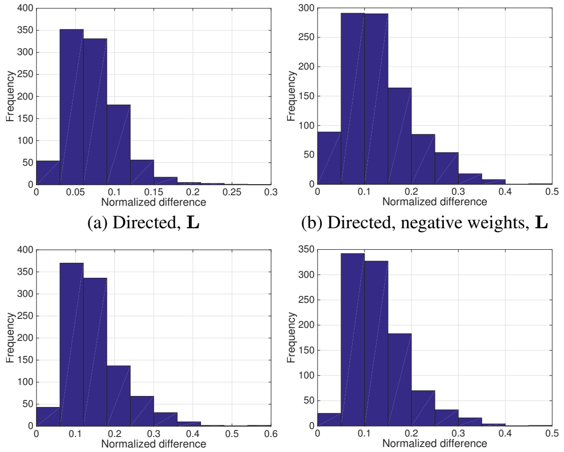
Directed network. In asymmetric networks we can define different Laplacians [24, 25] depending on whether we focus, e.g., in incoming connections or outgoing connections. The definition that preserves the diffusion interpretation is the outgoing directed Laplacian $\mathbf{L}_{\text{out}} := \mathbf{D}_{\text{out}} - \mathbf{A}$, where the diagonals of $\mathbf{D}_{\text{out}}$ contain the out-degree of each node. The diffusion filtered signal in asymmetric graphs can then be written as

$$\mathbf{y} = \left(\mathbf{I} + \alpha\, \mathbf{L}_{\text{out}}\right)^{-1} \mathbf{x}. \tag{11}$$

The componentwise relationship induced by the corresponding diffusion process is identical to (4). The difference is that the effect of node j in the rate of change of the value at node i is not necessarily equal to the effect of i in the rate of change of the value at j. As in the case of symmetric networks we can defined a normalized diffusion filter by replacing $\mathbf{L}_{\text{out}}$ with the normalized Laplacian $\tilde{\mathbf{L}}_{\text{out}} := \mathbf{D}_{\text{out}}^{-1}\mathbf{L}_{\text{out}}$ in (11).

Network with negative weights. In a network with negative weights, the positive weights encode affinity, null weights encode indifference, and negative weights dissimilarity; see e.g., [26]. Negative values in the Laplacian cannot be allowed in Definition 1 because the associated dynamical system in (2) is unstable. To resolve this problem we define the absolute degree matrix $\overline{\mathbf{D}}$ with diagonal elements $\overline{D}_{ii} = \sum_{j\in\mathcal{N}_i} |A_{ij}|$, [27]. The corresponding absolute Laplacian is defined as $\overline{\mathbf{L}} := \overline{\mathbf{D}} - \mathbf{A}$ and the diffusion filtered signal in networks with negative weights is defined as

$$\mathbf{y} = \left(\mathbf{I} + \alpha\, \overline{\mathbf{L}}\right)^{-1} \mathbf{x}. \tag{12}$$



(a) Directed, $\mathbf{L}$

(b) Directed, negative weights, $\mathbf{L}$

(c) Directed, negative weights, $\tilde{\mathbf{L}}$

(d) Negative weights, $\mathbf{L}$, $\|\cdot\|_2$

Fig. 1. (a), (b), (c) Histograms of the relative norm of the difference in the diffused signals $\|\mathbf{y}'-\mathbf{y}\|_\infty/\|\mathbf{E}\|_\infty\|\mathbf{x}\|_\infty$ for different network structures. (d) $\|\mathbf{y}'-\mathbf{y}\|_2/\|\mathbf{E}\|_2\|\mathbf{x}\|_2$ for networks with negative weights. The ratio is considerably lower than the theoretical upper bound of 1.

The generative diffusion process follows from substituting $\mathbf{L}$ by $\overline{\mathbf{L}}$ in (2). The resulting componentwise dynamics can be written as

$$\frac{dx_i(t)}{dt} = \sum_{j\in\mathcal{N}_i} \alpha \; |w_{ij}| \left(\text{sign}\left(w_{ij}\right) x_j(t) - x_i(t)\right). \tag{13}$$

The rate of change on i induced by neighbor j is governed by the magnitude of the weight between them. Edges with positive weights drive neighbor signals towards each other – as in regular diffusion – whereas the value at a node is driven to the additive inverse of the values at adjacent nodes when the shared edge has a negative weight. If the network contains negative weights and is also asymmetric, we replace $\overline{\mathbf{L}}$ with $\overline{\mathbf{L}}_{\text{out}} := \overline{\mathbf{D}}_{\text{out}} - \mathbf{A}$, where $\overline{\mathbf{D}}_{\text{out}}$ is a diagonal matrix containing the absolute out-degrees. A normalized version can be likewise defined.

4. STABILITY

The diffusion filtered signal depends on the underlying network structure through the associated Laplacian. It is therefore crucial to analyze how a perturbation of the underlying network affects the filtered signal. In this section we prove that for all the generalized diffusion filters defined in Section 3, a small network perturbation entails a small change in the diffused signal. We consider perturbations of a given network as noise added to its edge weights, and consequently quantify the network perturbation as the matrix p-norm of the difference between the Laplacians of the original and the perturbed networks. We focus our analysis on the most frequently used p-norms where $p \in \{1, 2, \infty\}$. We formally state our result next.

Proposition 1 *Given the unnormalized (normalized) Laplacian matrix* $\mathbf{L}$ *(*$\tilde{\mathbf{L}}$*) associated with any graph* $\mathcal{G}$*, potentially directed and with negative edge weights, and a diffusion rate* α*, for a bounded signal* $\mathbf{x}$ *on the network with* $\|\mathbf{x}\|_\infty \leq \gamma$*, if we perturb the network resulting in the Laplacian* $\mathbf{L}' = \mathbf{L} + \mathbf{E}$ *(*$\tilde{\mathbf{L}}' = \tilde{\mathbf{L}} + \mathbf{E}$*) with the difference* $\mathbf{E}$ *such that* $\|\mathbf{E}\|_\infty \leq \epsilon < 1$*, the difference between the diffused signal* $\mathbf{y}'$ *based on the perturbed network and the diffused signal* $\mathbf{y}$ *based on the original network is bounded by*

$$\left\|\mathbf{y}' - \mathbf{y}\right\|_\infty \leq \gamma\epsilon + o(\epsilon) < \gamma + o(1). \tag{14}$$

Moreover, if the original Laplacian $\mathbf{L}$ *(*$\tilde{\mathbf{L}}$*) is symmetric, then the above statement is also true when* $\|\cdot\|_\infty$ *is replaced by* $\|\cdot\|_1$ *or* $\|\cdot\|_2$*.*

Proof: See [28]. ■

Proposition 1 presents a bound for the difference between the diffusion filtered signals computed based on an original graph and its perturbed version. This bound depends on the original energy of the signals $\|\mathbf{x}\|_\infty$ as well as on the magnitude of the perturbation $\|\mathbf{E}\|_\infty$. Since the bound in (14) contains high order terms, we may divide (14) by ϵ and compute the limit as the perturbation vanishes

$$\lim_{\epsilon\to 0} \frac{\|\mathbf{y}' - \mathbf{y}\|_\infty}{\epsilon} \leq \gamma, \tag{15}$$

entailing that, for small perturbations, the difference in the diffused signals grows linearly. When establishing the underlying network in real-world applications, edge weights contain inherent errors. Proposition 1 ensures that diffused signals are robust to these minor perturbations.

To illustrate stability, we construct directed networks with edge weights uniformly picked either from $[0, 1]$ or $[-1, 1]$, the latter used to generate signed networks. We perturb these networks by adding to each edge a random weight uniformly picked from $[-0.05, 0.05]$, and then consider the diffused versions of a graph signal $\mathbf{x}$ – also randomly generated – with respect to the perturbed graph and the original graph, for both Laplacians $\mathbf{L}$ and $\tilde{\mathbf{L}}$. To verify Proposition 1 for the 2-norm $\|\cdot\|_2$, we also examine diffusion on symmetric networks. In Figure 1 we plot the histogram for 1000 network perturbations of the norm of the difference in the diffused signals $\mathbf{y}$ and $\mathbf{y}'$, normalized by both the norm of the perturbation ϵ and the norm of the graph signal γ, for every setting analyzed. It is immediate that all perturbations are below the theoretical upper bound of 1 [cf. (15)] by a considerable margin. This stability property is of practical relevance, as we present in the next section.

5. RECOMMENDATION SYSTEMS

Rating prediction, i.e., estimating a given user rating for a given item, is at the core of recommendation systems. Novel approaches to solve such problems, traditionally solved by machine learning tools, have been proposed recently [29, 30]. In this section, we illustrate a signal processing perspective for rating prediction and show how diffusion filtering can be used to improve the performance of recommendation systems.

Usually, ratings are considered in the form of a two dimensional matrix $\mathbf{R} = \mathbb{R}^{|\mathcal{U}|\times|\mathcal{I}|}$ for the set of users $\mathcal{U}$ and the set of items $\mathcal{I}$. Not every entry of $\mathbf{R}$ is known and the objective of the problem is to predict missing entries of interest, i.e. to obtain predictions $\hat{R}_{u,i}$ for user u and item i, from known entries $(v, j) \in \mathcal{K}$ with $\mathcal{K}$ denoting the set of all known entries. Two main techniques have been used to solve the prediction problem: matrix completion [31] and collaborative filtering (CF) [32]. Matrix completion looks for a low rank matrix $\hat{\mathbf{R}}$ such that $\hat{R}_{u,i} = R_{u,i}$ for all known entries $(u, i) \in \mathcal{K}$. The application of Laplacian smoothing in movie recommendation has been evaluated in [33] where it is considered in matrix completion. In this section, we apply the diffusion filtering in collaborative filtering. In *user-based* CF, given a certain item i, the recommendation system computes prediction $\hat{R}_{u,i}$ as the weighted sum of the ratings $R_{v,i}$ evaluated by different users v for the same item i, i.e.

$$\hat{R}_{u,i} = \frac{\sum_{v\in\mathcal{U}\setminus\{u\}} s(u, v) R_{v,i}}{\sum_{v\in\mathcal{U}\setminus\{u\}} s(u, v)}, \tag{16}$$

where $s(u, v)$ denotes a similarity or proximity between users u and v. In this paper, we consider cosine similarities

$$s_{\cos}(u, v) = \frac{\sum_{i\in\mathcal{I}} R_{u,i} R_{v,i}}{\sqrt{\sum_{i\in\mathcal{I}} R_{u,i}^2}\sqrt{\sum_{i\in\mathcal{I}} R_{v,i}^2}} = \frac{\mathbf{R}_u^T \mathbf{R}_v}{\|\mathbf{R}_u\|_2 \|\mathbf{R}_v\|_2}, \tag{17}$$

where $\mathbf{R}_u \in \mathbb{R}^{|\mathcal{I}|}$ is the vector of all ratings for user u and similarly for v. Since $s_{\cos}(u, v)$ computes the cosine of the angle between $\mathbf{R}_u$

	unnormalized Laplacian				normalized Laplacian				k-NN	matrix completion
	cosine	adj. cosine	Pearson	usage	cosine	adj. cosine	Pearson	usage		
user-based CF	0.8995	0.8997	0.9008	0.9042	0.8934	0.8937	0.8929	0.9028	0.9014	0.8981
item-based CF	0.8851	0.8856	0.8859	0.8867	0.8820	0.8817	0.8813	0.8825	0.8881	

Table 1. Global RMSEs for different recommender systems. Diffusion filtering, as a preprocessing step, improves the recommendation performance of collaborative filtering.

and $\mathbf{R}_v$, it is contained between -1 and 1, where larger positive values indicate stronger similarities between u and v and the opposite is true for negative values. In (17), to overcome the problem of having missing values in $\mathbf{R}$, we set all missing values to 0 and adjust the values for known ratings in $\mathbf{R}_u$ as $R_{u,i} := R_{u,i} - (\bar{R} + \mu_u), \forall (u,i) \in \mathcal{K}$ where $\bar{R}$ is the average over all known ratings $\bar{R} = \sum_{u,i:(u,i)\in\mathcal{K}} R_{u,i}/|\mathcal{K}|$ and μ_u is the user bias defined as

$$\mu_u = \frac{\sum_{i\in\mathcal{I}:(u,i)\in\mathcal{K}} \left(R_{u,i} - \bar{R}\right)}{|\{i \in \mathcal{I} : (u,i) \in \mathcal{K}\}|}. \tag{18}$$

In the above normalization, missing values in $\mathbf{R}$ are set to the average taste of each user and known ratings are adjusted to compensate for the fact that some users are more generous than others. Besides resolving missing values, the mean centering also favors the performance of recommendation systems [32, 34]. By contrast, in *item-based* CF, given a user u, the system predicts rating $\hat{R}_{u,i}$ as the weighted sum of the ratings $R_{u,j}$ evaluated by the same user u for different items j [cf. (16)]

$$\hat{R}_{u,i} = \frac{\sum_{j\in\mathcal{I}\setminus\{i\}} s(i,j) R_{u,j}}{\sum_{j\in\mathcal{I}\setminus\{i\}} s(i,j)}. \tag{19}$$

Again, in this paper we use cosine similarity as measure $s(i,j)$, i.e., the analogous to (17) but between items instead of users. One usually restricts attentions to the k users – in user-based CF – or items – in item-based CF – that are the most similar to the user or item of interest in the weighted sum. Such restrictions yield user- or item-based k-Nearest Neighbor (k-NN) CF.

In contrast with the conventional formulation, we give an interpretation of the problem from a signal processing perspective. For each user u, the rating vector $\mathbf{R}_u$ can be viewed as a graph signal defined on top of a common network $\mathcal{G}_{\text{item}}$ describing proximities between items. Under such setup, users u and v should be similar if they assign similar ratings to similar movies, even when the items rated by them do not coincide. This interpretation motivates the use of diffusion filtering to smooth rating vectors $\mathbf{R}_u$ before evaluating the cosine similarities. We refer to this formulation as item-based-diffusion user-based k-NN.

We consider 4 different ways to construct proximity networks between movies to encode similarities between them. First, we can compute the edge weight between two movies i and j as its cosine similarity $s_{\cos}(i,j)$ [cf. (17)]. Second, we can find edge weights through adjusted cosine similarity [35] by normalizing average ratings for each movie. Third, we may utilize Pearson correlation [36] by restricting attention to users who rate both movies i and j. All these three constructions – cosine, adjusted cosine, and Pearson – are based on comparing the rating profiles of two items. The intuition is that movies are similar if their reviews by the same users are similar. The resulting networks would be symmetric and, potentially, have edges with negative weights. Another way to construct the network is to consider the proportion of users who rate both movies relative to the users who rate only one of them

$$s_{\text{usage}}(i,j) = \frac{|\{u \in \mathcal{U} : (u,i) \in \mathcal{K}, (u,j) \in \mathcal{K}\}|}{|\{u \in \mathcal{U} : (u,i) \in \mathcal{K}\}|}. \tag{20}$$

The above formulation is based on the idea that two movies are similar if the set of users that rate them are similar, thus, yields directed networks since different movies enjoy different levels of popularity represented by the denominator in (20). In all constructions, edges are removed if their weights are smaller than a threshold in absolute values.

The rating prediction problem can also be considered from the dual domain by switching the roles of items and users. For each item i, its ratings $\mathbf{R}_i \in \mathbb{R}^{|\mathcal{U}|}$ evaluated by different users can be considered as a graph signal defined on top of a common network $\mathcal{G}_{\text{user}}$ describing proximities between users. Under such setup, items i and j should be similar if they are given similar ratings by users with alike tastes, even if there is no user that rated both of them. We therefore consider smoothing the ratings $\mathbf{R}_i$ utilizing a similarity network between users before computing the cosine similarity between items in collaborative filtering, and call this method user-based-diffusion item-based k-NN.

To illustrate how diffusion filtering can facilitate the similarity computation and improve the performance of recommendation systems, we consider the MovieLen-100k dataset [37] containing 100,000 ratings from 943 users on 1,682 movies. We perform 5-fold cross validation by partitioning the users into 5 random sets. For each cross validation round, 10 randomly selected ratings from each of the users in a given fold are withheld and kept as the test set, with all remaining ratings forming the training set. As a performance metric we use the global root mean squared error (RMSE), averaged across the 5 different testing sets. Matrix completion and k-NN CFs without diffusion filtering are utilized as benchmark algorithms. In diffusion filtering, we construct different underlying networks based only on the available training set.

The global RMSEs for different methods averaged across 3 different 5-fold partitions of the data are summarized in Table 1. Collaborative filtering with 8 different diffusion filters – 4 ways to construct networks and 2 choices of Laplacians – all improve the RMSE values except for the user-based CFs with usage networks. Also, user-based CFs outperform matrix completion methods when preprocessed with normalized diffusion filtering. The almost 1% improvement in RMSE – from 0.9014 to 0.8929 and from 0.8881 to 0.8813 – is substantial in rating prediction problems, considering that the benchmark algorithms are highly optimized [32]. The improvements increase to $1.5\% - 4\%$ when networks are constructed from all ratings, as opposed to only those in the training set. The results presented are robust for a wide range of neighbors k, diffusion rates α, and thresholding parameters used in constructing the networks. The only diffusion filter that increases RMSE is user-based CF with usage networks, partly due to the fact that (20) is based only on the *number* of ratings and not on the *content* of these. This problem can be solved by considering positive ratings separately from negative ratings. It is also observed that normalized Laplacian works better than unnormalized Laplacian. Part of the reason is because some movies or users tend to be isolated – few weak connections – in the constructed networks after edge weight thresholding. Therefore, diffusion with unnormalized Laplacian cannot effectively act on these isolated nodes and the normalized Laplacian achieves a better performance (cf. Section 3).

6. CONCLUSION

We defined diffusion filtering as a smoothing method for signals supported on networks. The method relies on the temporal heat map induced by the diffusion of signals across the network and evaluates the accumulated temperatures across time. We showed that diffused signals are stable with respect to perturbations in the underlying network and demonstrated that diffusion filtering can improve the performance of recommendation systems.

7. REFERENCES

[1] D. Bu, Y. Zhao, L. Cai, H. Xue, X. Zhu, H. Lu, J. Zhang, S. Sun, L. Ling, and N. Zhang, "Topological structure analysis of the protein–protein interaction network in budding yeast," *Nucleic acids research*, vol. 31, no. 9, pp. 2443–2450, 2003.

[2] E. Lieberman, C. Hauert, and M. Nowak, "Evolutionary dynamics on graphs," *Nature*, vol. 433, no. 7023, pp. 312–316, 2005.

[3] M. E. J. Newman, "Finding community structure in networks using the eigenvectors of matrices," *Phys. Rev. E*, vol. 74, p. 036104, 2006.

[4] J. Kleinberg, "Authoritative sources in a hyperlinked environment," *J. ACM*, vol. 46, no. 5, pp. 604–632, Sep. 1999.

[5] D. Shuman, S. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 83–98, Mar. 2013.

[6] R. Mittler, S. Vanderauwera, M. Gollery, and F. V. Breusegem, "Reactive oxygen gene network of plants," *Trends in Plant Science*, vol. 9, no. 10, pp. 490 – 498, 2004.

[7] O. Sporns, *Networks of the Brain*. MIT press, 2011.

[8] A. Luikov, *Analytical heat diffusion theory*. Academic press, New York, 1968.

[9] E. Eckert and R. Drake, *Analysis of heat and mass transfer*. Hemisphere Publishing; New York, NY, 1987.

[10] R. I. Kondor and J. Lafferty, "Diffusion kernels on graphs and other discrete input spaces," in *ICML*, vol. 2, 2002, pp. 315–322.

[11] A. Szlam, M. Maggioni, and R. Coifman, "Regularization on graphs with function-adapted diffusion processes," *J. Mach. Learn. Res.*, vol. 9, pp. 1711–1739, 2008.

[12] J. A. F. R. Olfati-Saber and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, 2007.

[13] J. C. Dittmer, "Consensus formation under bounded confidence," *Nonlinear Analysis*, vol. 47, 2001.

[14] F. Chung, *Spectral graph theory*. American Mathematical Soc., 1997, vol. 92.

[15] R. A. Horn and C. R. Johnson, *Matrix analysis*. Cambridge university press, 2012.

[16] A. J. Smola and R. Kondor, "Kernels and regularization on graphs," in *Learning theory and kernel machines*. Springer, 2003, pp. 144–158.

[17] X. Dong, D. Thanou, P. Frossard, and P. Vandergheynst, "Laplacian matrix learning for smooth graph signal representation," in *Proc. Int. Conf. Acoustics Speech Signal Process*, Brisbane, Australia, April 19-24 2015, pp. 3736–3740.

[18] A. Sandryhaila and J. Moura, "Discrete signal processing on graphs," *IEEE Trans. Signal Process.*, vol. 61, no. 7, pp. 1644–1656, Apr. 2013.

[19] S. Segarra, A. G. Marques, G. Leus, and A. Ribeiro, "Interpolation of graph signals using shift-invariant graph filters," in *European Signal Process. Conf. (EUSIPCO)*, Nice, France, Aug. 31 - Sept. 4, 2015.

[20] ——, "Reconstruction of graph signals through percolation from seeding nodes," *arXiv preprint arXiv:1507.08364*, 2015.

[21] F. Zhang and E. R. Hancock, "Graph spectral image smoothing using the heat kernel," *Pattern Recognition*, vol. 41, no. 11, pp. 3328–3342, 2008.

[22] A. Kheradmand and P. Milanfar, "A general framework for regularized, similarity-based image restoration," *Image Processing, IEEE Transactions on*, vol. 23, no. 12, pp. 5136–5151, 2014.

[23] U. von Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, 2007.

[24] F. Chung, "Laplacians and the cheeger inequality for directed graphs," *Annals of Combinatorics*, vol. 9, no. 1, pp. 1–19, 2005.

[25] Y. Li and Z.-L. Zhang, "Random walks on digraphs, the generalized digraph laplacian and the degree of asymmetry," in *Algorithms and Models for the Web-Graph*, ser. Lecture Notes in Computer Science, R. Kumar and D. Sivakumar, Eds. Springer Berlin Heidelberg, 2010, vol. 6516, pp. 74–85.

[26] F. Harary, "On the notion of balance of a signed graph," *The Michigan Mathematical Journal*, vol. 2, no. 2, pp. 143–146, 1953.

[27] J. Kunegis, S. Schmidt, A. Lommatzsch, J. Lerner, E. De Luca, and S. Albayrak, "Spectral analysis of signed graphs for clustering, prediction and visualization," in *Proceedings of the SIAM Intl.Conf. on Data Mining*. Society for Industrial and Applied Mathematics, 2010, pp. 559–570.

[28] W. Huang, S. Segarra, J. Ma, and A. Ribeiro, "Diffusion filtering for signals supported on networks," *IEEE Trans. Signal. Process.*, vol. (submitted), 2015.

[29] J. Z. Sun, K. R. Varshney, and K. Subbian, "Dynamic matrix factorization: A state space approach," in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*. IEEE, 2012, pp. 1897–1900.

[30] S. K. Narang, A. Gadde, and A. Ortega, "Signal processing techniques for interpolation in graph structured data," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 5445–5449.

[31] J.-F. Cai, E. J. Candès, and Z. Shen, "A singular value thresholding algorithm for matrix completion," *SIAM Journal on Optimization*, vol. 20, no. 4, pp. 1956–1982, 2010.

[32] M. D. Ekstrand, J. T. Riedl, and J. A. Konstan, "Collaborative filtering recommender systems," *Foundations and Trends in Human-Computer Interaction*, vol. 4, no. 2, pp. 81–173, 2011.

[33] V. Kalofolias, X. Bresson, M. Bronstein, and P. Vandergheynst, "Matrix completion on graphs," *arXiv preprint arXiv:1408.1717*, 2014.

[34] M. D. Ekstrand, M. Ludwig, J. A. Konstan, and J. T. Riedl, "Rethinking the recommender research ecosystem: reproducibility, openness, and lenskit," in *Proceedings of the fifth ACM conference on Recommender systems*. ACM, 2011, pp. 133–140.

[35] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proceedings of the 10th international conference on World Wide Web*. ACM, 2001, pp. 285–295.

[36] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl, "An algorithmic framework for performing collaborative filtering," in *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 1999, pp. 230–237.

[37] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "Grouplens: an open architecture for collaborative filtering of netnews," in *Proceedings of the 1994 ACM conference on Computer supported cooperative work*. ACM, 1994, pp. 175–186.